Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Technology 3 (2012) 167 – 177

Procedia Technology

The 2012 Iberoamerican Conference on Electronics Engineering and Computer Science

Semantic Discovery of Web Services through Social Learning

Sajib Kumar Mistry[a,*], Mosaddek Hossain Kamal[b], Dilip Mistry[c]

[a]*Computer Science and Engineering, University of Dhaka, Dhaka-1000, Bangladesh*
[b]*Electrical and Computer Engineering, Concordia University, Canada*
[c]*Transportation and Logistics, North Dakota State University, USA*

Abstract

The increasing numbers of web services impose automatic discovery process in Service Oriented Architecture (SOA). But the existing SOA enables only syntactic discovery which produces coarse irrelevant results or sometimes no results. Different researches challenge this problem by introducing semantic discovery process in SOA to enable relevant and desired search results. These research outcomes cannot discover services efficiently which are created independently with different knowledge bases. To overcome these problems, a new architecture of SOA is proposed which incorporates a new adaptive technique called social learning that improves service provider's domain ontology from service consumer's concept contributions and thus eventually makes the service more semantically discoverable. The proposed architecture contains new similarity measure and automatic merging algorithms on weighted ontology. From mathematical reasoning it is induced that the proposed architecture reduces overlapping concepts and thus more relevant discovery results are ensured. To test the proposed architecture's performance, a prototype of Universal Description Discovery and Integration (UDDI) is implemented and a simulation is conducted with real data set of OWL-S Technical Chart (OWLS-TC). About 67% noise responses from syntactic search (N-Gram String Distance algorithm) are reduced in the proposed architecture. The results also illustrate the proposed architecture's capabilities of concept learning and so about significant improvement in service discovery after a few social concept contributions.

Keywords: Distributed Discovery in SOA, Semantic Web Service, Weighted Ontology Engineering, Cognitive Machine Learning, Social Contributions and QOS in SOA

1. Introduction

The web has been evolving into the world's largest distributed system for sharing information. The growth of web and wide acceptance of web protocols in different systems makes the hype of Service Oriented Architecture (SOA) popularly known as web service [1]. In SOA, a service provider exposes its services in standard Web Service Description Language (WSDL). From the WSDL specification the service consumer can make software clients to consume the service. The interacting messages are carried in SOAP as payload of different web protocols like HTTP, SMTP etc. The service provider can register their services

*Tel: +8801720254208
Email addresses: sajib.kumar.mistry@gmail.com (Sajib Kumar Mistry), tushar@cse.univdhaka.edu (Mosaddek Hossain Kamal), dilip.mistry@my.ndsu.edu (Dilip Mistry)

2212-0173
doi:10.1016/j.protcy.2012.03.018

in Universal Discovery, Description and Integration (UDDI). The UDDI gives API for searching web services for the service requester. The architecture of web services can be treated as a great application field of Semantic Web [2]. Sir Timothy John "Tim" Berners-Lee (director of W3C and the creator of WWW) visualized semantic web more powerful than traditional web where the machines can understand the meaning of the information and can process the information automatically. But the standard SOA architecture does not provide semantic interoperability in any of its components . WSDL does not specify about what the service does (Service Domain and UDDI only provides service for syntactical discovery of web services. It restricts user agents to find desired service. The syntactic search produces either noisy irrelevant results or no results. But if we can annotate WSDL with semantic knowledge and make UDDI capable of semantic search then complex discovery of web services is possible.

Over the years different researchers have approached differently to add semantic search capability in SOA. Some researches only describe the service semantically [3]. But they do not give semantic searching capabilities in UDDI. Some researchers add semantic searching in UDDI but restrict the architecture in non-scalable single point searching methodologies [4]. Besides some researchers restrict the service requester and provider to follow an imaginary world concept [5]. These researches restrict SOA to be more flexible semantically. From this paper viewpoint semantically enabled SOA should follow the following characteristics.

- it should allow the semantic description of services within the concrete service description.
- it should allow scalable searching with different service mechanisms at different endpoints. Thus different searching algorithms can be used at different endpoints.
- it should allow the service requester and provider to create service ontology in a distributed manner. It is trivial that the chance of prior contact between them is very low. So they should create service description independently.

In this paper a new architecture of SOA is proposed that follows the above characteristics and is capable of semantic search. As the service domain ontology is completely distributed, the proposed architecture imposes a social form of improving ontology of the registered services. The main motto is that if service requester consumes a service, then it may contribute to improve the service provider's ontology. Besides there is no world ontology in the proposed architecture, so there should be a mechanism to update provider's ontology over time through learning from internet. In this paper this kind of learning is called social learning and a new service discovery process is proposed based on this social learning. The Fig 1(b) illustrates the concept of social learning. To carry this social learning new ontology matching and merging algorithm are introduced since existing ones can not meet the exact requirement. The rest of the paper is divided into sections where section 2 illustrates the new architecture; section 3 is describing new semantic matching algorithm; The social learning process through semantic merging of ontologies is in section 4 and finally simulation results are in section 5.

2. A New Social Architecture of SOA for Semantic Discovery

The architectures provided by J. Phatak et al. [6], Ivan M et al. [7], Bo Zhou et al. [8], Kim Christensen et al. [9], Naveen Srinivasan [10] etc. provide semantic discovery. Some of them are closed system whether some of them are distributed. In ideal case we need a distributed system to protect single point failure. Some of the existing architectures do not provide representation heterogeneity. That means to represent the concept in different formats. But again it is a desired option. Popular systems do not impose a certain format so that it can find popular format through recommendation. The main lack of the existing discussed architectures is to perform poor with services created with different knowledge bases. For example, assume there is no knowledge base to reason that "Toyota is a car". So, all the existing discussed architectures will return null matching in searching "Toyota" in "car" concept all the time. But the knowledge base can be improved over time through adaptive learning techniques. Thus further queries will not return null matching. Thus the overall searching results get improved. Another important lack in present architectures remains for

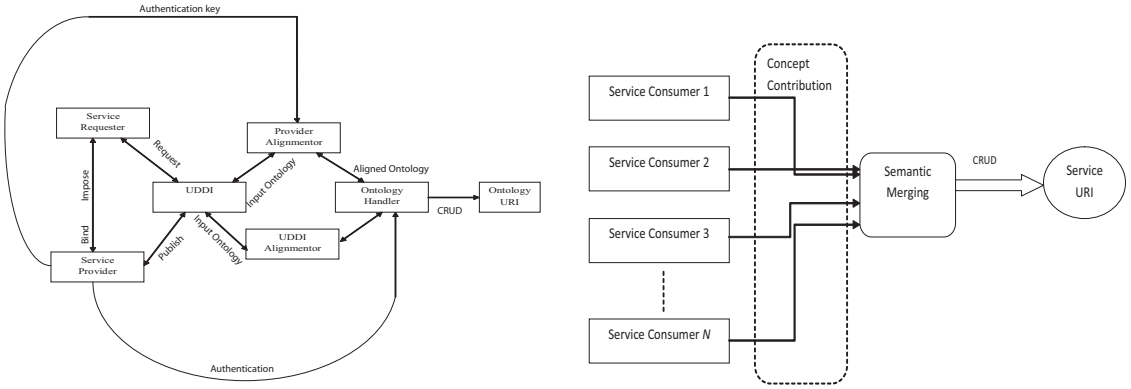


Fig. 1: (a) Proposed architecture for semantic discovery of web services in SOA; (b) The concept of Social learning

not having matching technique for weighted ontology. Weighted ontology reduces the concept overlapping and thus more accurate result is ensured. So we need an architecture that incorporates adaptive learning of weighted concepts along with representation heterogeneity and distributed nature.

2.1. The Improved Architecture of SOA for Semantic Discovery of Web Services

SOA already consists of three entities: service provider, service requester and UDDI. Here two new actors are needed:

- Alingmentor: The task of the alignmentor is to match ontologies with specified algorithm and send the threshold value to the UDDI. It also merges the ontologies and send the result to OntologyHandler.
- OntologyHandler: The main task of OntologyHandler is to perform CRUD(create, read, update and delete) operation in ontology URI. It can use REST methods to communicate with ontology URI.

The framework prototype is given in Fig 1(a). In the architecture service requester requests to the UDDI with the ontology of its interest with a lower threshold value for matching. The UDDI will return the service endpoints of the all matched services. The UDDI supports another type of request: the merge request. In this request the requester provides specific service provider's key to merge the ontology. The service publisher will publish its service with the domain ontology URI. It also specifies whether it has own alignmentor or it will use the default UDDI alignmentor. For every service which has own alignmentor, UDDI will send the requested ontology to Provider Alignmentor if the alignmentor supports the type of ontology. Provider alignmentor returns the matched threshold value. UDDI can also send ontology for merging with authentication key. Besides, UDDI to UDDI Alignmentor conversation will happen only when UDDI does not find Provider Alignmentor. The conversations are alike in UDDI to Provider Alignment and vice versa. The provider will make the authentication between the alignmentor and handler. So, if the alignmentor wants to update the ontology, ontologyHandler allows the operation.

3. A New Algorithm for Semantic Matching Process of Web Service Descriptions

The success of proposed architecture depends a lot on efficient semantic matching and merging of ontologies. Semantic matching can then be defined as the following problem: Given two ontologies O , O' and an alignment A between them, semantic matching process computes A' which is the set of mapping elements M where $e, e' \in A$. Structure based mapping takes into account the graph like representation of the ontology

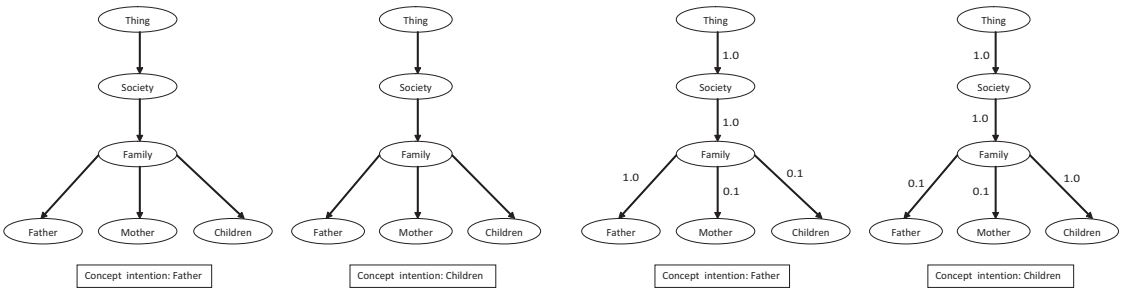


Fig. 2: Two different intended concepts with same (a) non weighted structure; (b)weighted structure

(Cupid, S-Match, COMA etc) [11]. These algorithms for semantic matching only return the mapped set of Alignmentor. But it does not provide the overall similarity matching of the concepts. To match overall similarity non-weighted ontologies cannot meet the criteria. As two different intended concepts can have same structural representation, non-weighted ontologies may return same similarity metrics. In Fig 2(a) there are two ontologies with same structure. Here, one intends to describe the "Concept: Father" and other intends to describe "Concept: children". If the non weighted structure is provided then any similarity matching process may give the same result for both of them. So to distinguish concepts of intention we need to concentrate more on the desired labels. Thus we need to introduce weight in the ontology structure. The hypothesis is that : "The more we find match in high weighted links the more similarity is between the concepts". The Fig 2(b) describes that there are high weighed values near the intended concept concentration. The following section describes the weighted matching process in details.

3.1. Proposed Semantic Similarity Measure Algorithm for Weighted Ontology

The proposed Semantic Similarity Measure Algorithmalgorithm has two phases. In off-line phase two matrices **GenericSimilarity[**N**][**N'**]** and **StructureSimilarity[**N**][**N'**]** are computed using existing common ontology (for example, WordNet knowledge base etc.) and existing generic or architectural solutions (for example, P&S, S-Match, COMA etc.). The matrix **GenericSimilarity[**N**][**N'**]** contains values for each $(n1, n2)$ where $n1 \in N$ and $n2 \in N'$ irrespective of the structure of the ontology. On the other side,The matrix **StructureSimilarity [**N**][**N'**]** contains values for each $(n1, n2)$ where $n1 \in N$ and $n2 \in N'$ respective of the structure of the ontology. Link similarity is not considered in existing structural similarity measures in off-line phase. But it returns important matching where the structural and generic similarity perform low due to huge difference between input ontology knowledge bases. These huge differences are generally produced for using more instance in ontology description. As normally similarity mapping between instances are low due to lack of generic term, both structural and generic similarity perform poor matching. If two trees have same link property, though there is no generic and structural similarity between the concepts, only strong link object property match can impose more similarity than expected. As service provider's concept and requester's concept normally tend to use different knowledge base, link similarity may give the higher result where other similarities perform low. So in on-line phase, to calculate the similarity between overall tree there needs a mapping between edges. For a edge $e(n1, n2, w)$ in ontology O and edge $e'(n1', n2', w)$ three kinds of similarity can be assumed:

- Generic Similarity: For linked nodes the following equation can be used.

$$GenericSimilarity = \alpha_{Gen}.GenericSimilarity[n1][n1'] + \beta_{Gen}.GenericSimilarity[n2][n2'] \quad (1)$$
$$where,\ \alpha_{Gen} + \beta_{Gen} = 1.0$$

The variables α_{Gen} and β_{Gen} is depended on system environment set up by the administrator.

- Structural Similarity: For linked nodes the following equation can be used.

$$StructuralSimilarity = \alpha_{Str}.StructureSimilarity[n1][n1^{'}]+\beta_{Str}.StructureSimilarity[n2][n2^{'}] \quad (2)$$
$$where,\ \alpha_{Str}+\beta_{Str}=1.0$$

The variables α_{Str} and β_{Str} is depended on system environment set up by the administrator.

- Link Similarity: The link similarity between edge e and $e^{'}$ can be computed as the following

$$LinkSimilarity = sim_{tvr'}(m1,m2);\ where\ m1 \in e\ and\ m2 \in e^{'}[element\ level\ similarity] \quad (3)$$

- Overall Similarity: using equations 1, 3, 3 the overall similarity index can be computed as the following equation.

$$Similarity = \alpha.StructuralSimilarity + \beta.GenericSimilarity + \gamma.LinkSimilarity \quad (4)$$
$$where\ ,\ \alpha+\beta+\gamma=1.0$$

The γ can influence the α and β on the value of *LinkSimilarity*. Thus γ in equation 4 has relation with α and β. Assuming the proportional relation with linksimilarity and overall similarity the following equations can be introduced:

$$\gamma = LinkSimilarity * 0.5$$
$$\alpha = \alpha - \gamma/2; \beta = \beta - \gamma/2; \quad (5)$$

3.1.1. Calculation of Weighted Similarity Index among Links

From equation 4 and 5 the similarity measure of the edges are calculated. But it does not include the weight of links. The assumption is that an edge may give high similarity with other's ontology's edge if the difference between the link weights is minimum. The hypothesis is that to find similarity index with high values; edges of the ontologies should match more on high weights as it represents that the concentration of concepts within same area. This assumption brings the following equation 6 using *Similarity* from equation 4.

$$WeightedSimilarity = \frac{Similarity \times min(e(n1,n2,w), e^{'}(n1^{'},n2^{'},w^{'}))}{max(e(n1,n2,w), e^{'}(n1^{'},n2^{'},w^{'}))} \quad (6)$$

3.2. Formal Algorithm for Similarity Index between Weighted Ontologies

The algorithm for similarity measure of concepts is given in algorithm 1.

3.3. How Proposed Algorithm Performs Better Than Non-Weighted Matching Algorithms

The proposed similarity measure algorithm deals with weighted ontology. Non weighted ontology can be viewed as weighted ontology with maximum weight equally distributed on every links. Thus matching non-weighted ontologies is like matching with syntactic capabilities. Due to the maximum weight on every link of a non-weighted ontology $C_{non-weighted}$ can be viewed as a more general representation of weighted ontology $C_{weighted}$. The equation 7 illustrates the situation. Using rules of F. Giunchiglia et al. [12] for converting ontology relation to propositions; the equation 7 can be written as equation 8.

$$C_{weighted} \sqsubseteq C_{non-weighted} \quad (7)$$

$$C_{weighted} \rightarrow C_{non-weighted} \quad (8)$$

Suppose there are N service provider. Each provider has a service domain concept C in both weighted and non-weighted way. Let $C\_Over_{non-weighted}$ as the overlapping concept for non weighted ontologies and

Algorithm 1 Semantic Similarity Measure of Concepts

Input: Two ontology $O[N, E, W]$ and $O^{'}[N^{'}, E^{'}, W^{'}]$ where N is node, e is edge and W is weight
Output: Similarity Measure S where $S \in [0, 1]$

```
count ← 0
S ← 0
for each edge e(n1, n2, w) in O do
    max ← 0
    for each edge e'(n1', n2', w') in O' do
        if Weighted Similarity(e, e') > max then
            max ← Weighted Similarity(e, e')
        end if
    end for
    S ← S + max
    count ← count + 1
end for
S ← S/count
return S
```

$C\_Over_{weighted}$ as the overlapping concept for non weighted ontologies. They can be written as in equation 9 and equation 10.

$$C\_Over_{weighted} = (C1_{weighted} \cap C2_{weighted} \cap C3_{weighted} \cap C4_{weighted} \cap ..... \cap CN_{weighted}) \tag{9}$$

$$C\_Over_{non-weighted} = (C1_{non-weighted} \cap C2_{non-weighted} \cap C3_{non-weighted} \cap ..... \cap CN_{non-weighted}) \tag{10}$$

Using equations in 8, 9 and 10 we get the following reasoning.

$$(C1_{weighted} \cap .. \cap CN_{weighted}) \rightarrow (C1_{non-weighted} \cap .. \cap CN_{non-weighted}) \Rightarrow C\_Over_{weighted} \rightarrow C\_Over_{non-weighted} \tag{11}$$

Also from equation 6 we get the following reasoning:

$$\begin{aligned} &\frac{Similarity \times min(e, e^{'})}{max(e, e^{'})} \leq \frac{Similarity \times max(e, e^{'})}{max(e, e^{'})} \\ &\Rightarrow \; WeightedSimilarity \leq Similarity \end{aligned} \tag{12}$$

The equation 11 states that overlapping concepts in non-weighted ontologies is more general than overlapping concepts in weighted ontology. Thus weighted ontology reduces the probability of overlapping concepts and less response index in matching. Besides the weighted ontology from service requester make the request more specific. So, only accurate and desired results will get higher matching index. Further the equation 12 ensures that weighted similarity is always less than or equal to the non-weighted similarity. So a higher response in proposed measure algorithm ensures noise free accurate data. Besides, the adaptive nature of equation 4 helps to find desired service on different circumstances. The coefficients α, β and γ represent weights on generic, structural and link similarity respectively. Now as it is stated that structural similarity performs poor in little knowledge base, by assigning $\alpha = 0$, $\beta = 0$ and $\gamma = 1$ we can find the similarity index of the whole link similarity. It is true for other coefficients also. Thus the adaptive generalized similarity measure algorithm on weighted ontology provides better accurate and relevant results than non-weighted similarity measure algorithms.

4. A New Algorithm for Social Learning on Web Service Descriptions Through Concept Alignment

Social learning is the process where if an individual becomes more social, more it can mix or share with others. It can be said in other words like if an individual can share itself with other individuals then

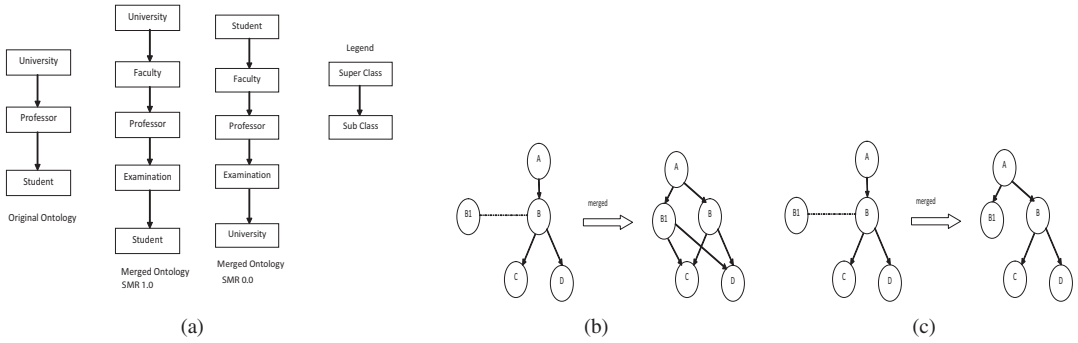


Fig. 3: (a) A structural mapping ratio; (b) Strong Merging;(c) Weak Merging

it becomes more social. The definition of social learning in the context of web service discovery can be like this : "*Social Learning for a web service description is the process of knowing about itself from service consumer's view point and updating own ontology according to it so that the similarity matching index would be high if the service stays active over time.*" So if a web service has weighted ontology $O[N, R, W]$, it will be updated over time using merging techniques. It concludes with the following hypothesis :

"If the similarity matching index for discovery ontology $O'[N', R', W']$ is M in time t_1 and M' in time t_2 then social learning imposes $M' > M$ if $t_1 > t_2$".

Social learning needs an efficient concept alignment process. Existing ontology merging algorithms (PROMT, SMART, ONION, MARFA [13]) do not deal with weighted ontology. As the proposed architecture depends on weighted ontology, a new alignment algorithm for weighted ontology is needed. Besides, existing merging tools do not incorporate any adaptive learning technique. Thus the new algorithm should also incorporate social learning. The existing merging algorithms also do not take the source ontology's semantic structure into consideration. But in social learning the contributor's ontology should preserve it's semantic meaning after merging. So we need a new merging algorithm for social learning on weighted ontology that preserves the semantics of service consumer's ontology. In later sections the new algorithm for concept alignment is discussed. It uses similarity measure index from algorithm 1.

4.1. A New Concept of Structural Mapping Ratio (SMR)

When ontology $O[N, R, W]$ is merged into $O'[N', R', W']$ each edge in $O[N, R, W]$ is decomposed from the original one and then merged into $O'[N', R', W']$. Structural mapping ratio is the measurement about how much the decomposed $O[N, R, W]$ retains its structure in new merged ontology $O''[N'', R'', W'']$. The structural mapping ratio (SMR) outputs in range [0, 1]. The equation for computing structural mapping ratio is given below in Equations 13 and 14:

$$Structural\ Mapping\ Ratio\ for\ a\ node,\ n = \frac{P'}{P}\ where, \tag{13}$$
$$P' = number\ of\ similar\ parents\ of\ n\ in\ merged\ ontology$$
$$P = number\ of\ parents\ of\ n\ in\ original\ ontology$$

$$Structural\ Mapping\ Ratio,\quad SMR = \frac{\sum_{n}^{N} Structural\ Mapping\ Ratio\ for\ n}{Number\ of\ nodes} \tag{14}$$

The Fig 3(a) depicts SMR 1.0 and 0.0.

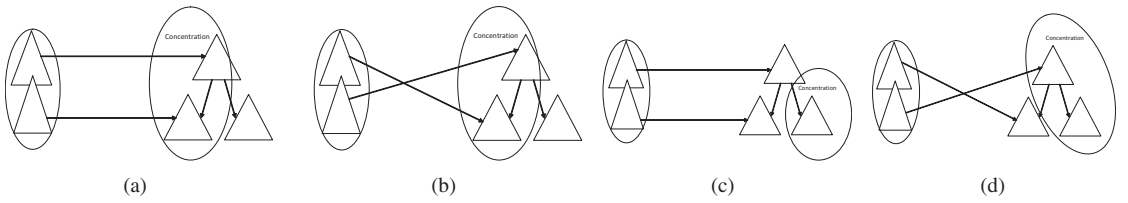


Fig. 4: (a) Both $Similarity(O, O^{'})$ and SMR are high;(b) $Similarity(O, O^{'})$ is high but SMR are low;(c) $Similarity(O, O^{'})$ is low but SMR are high; (d) Both $Similarity(O, O^{'})$ and SMR are medium

4.2. Hypothetical Scenarios in Weighted Ontology Merging

Suppose the ontology $O^{'}$ is getting merged with O and number of classes in $O^{'}$ is less than number of classes in O. For each class pair $(n1^{'}, n2^{'})$ in $O^{'}$ we get the best matched $(n1, n2)$ in O with similarity value S. The overall similarity between the concepts is $Similarity(O, O^{'})$. Besides by simply mapping each $(n1^{'}, n2^{'})$ in O to $(n1, n2)$ the Structure Mapping Ration(SMR) is computed. The following hypothetical scenario may arise based on the computed values.

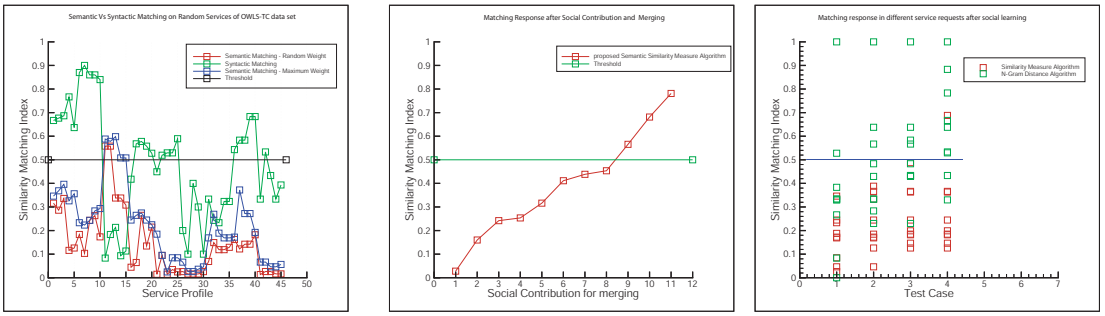
- When both $Similarity(O, O^{'})$ and SMR are high: This kind of scenario could occur when ontology $O^{'}$ is mapped with concentration points of ontology O preserving it's own meaning. It means that service provider's ontology consists most part of the consumer's ontology in it's core concept. The Fig 4(a) depicts the scenario.
- When $Similarity(O, O^{'})$ is high but SMR are low: This kind of scenario could occur when ontology $O^{'}$ is mapped with concentration points of ontology O but cannot preserve its own meaning. It means that service provider's ontology consists most part of the consumer's ontology in its core concept in a disperse way. The Fig 4(b) depicts the scenario.
- When both $Similarity(O, O^{'})$ and SMR are medium: This kind of scenario could occur when some parts of ontology $O^{'}$ is mapped with some of concentration points of ontology O and other parts are mapped with non concentrated parts of ontology O. It means that service provider's ontology consists some part of the consumer's ontology and vice versa. The Fig 4(d) depicts the scenario.
- When $Similarity(O, O^{'})$ is low and SMR are high: This kind of scenario could occur when ontology $O^{'}$ is mapped with other parts ontology O rather than the concentration points. It means that service provider's ontology consists most part of the consumer's ontology in its non-core concept. The Fig 4(c) depicts the scenario.
- When both $Similarity(O, O^{'})$ and SMR are low: This kind of scenario could occur when ontology $O^{'}$ is completely dissimilar from O in all aspects like structure and concepts at levels.

4.3. Introducing Strong and Weak Types of Merging

By analyzing the different case scenario, it is an assumption that different types of merging are needed in different hypothetical cases. These types differ in connectedness and weight distribution. There may be two types of connectedness: Strong and Weak and so there are two types of merging: Strong Merging and Weak Merging. Strong Merging and Weak Merging are illustrated in the Fig 3(b) and 3(c).

4.4. Assigning Weights to The Generated Links

Whether it is strong or weak merging, it is needed to assign new weights to the new links. It is the position of the node in the ontology that defines the weight. The effects of link weight in merging ontology only affects in new edges containing both nodes at each end. Now the confidence index in merging node can



(a) (b) (c)

Fig. 5: (a) Semantic Vs Syntactic Matching on Random Services of OWLS-TC data set; (b) Matching Response after social contribution and merging;(c) Matching Response in different service requests after social learning

be viewed from similarity measure table in similarity measure matching algorithm. Using this assumption the following equation is evolved:

$$\begin{aligned} &new\,weight,\ W^{'} = S \times W\ where, \\ &S = similarity\,measurement\,of\,the\,mapped\,link \\ &W = existing\,weight\,of\,the\,mapped\,link \end{aligned} \tag{15}$$

4.5. Formal Algorithm for Merging with Social Contribution

Using the discussions, assumptions and hypothetical cases, at first the nodes are sorted in ascending order according the number of parents including itself. The core reason for doing this is to place the more independent nodes in the ontology earlier than the dependent ones. Then for High SMR **"Strong Merging"** techniques are the trivial choice and **"Weak Merging"** techniques are used in medium SRM. In case of low SRM; normally it is a risk to go in automatic update but high similarity measure may impose **"Weak Merging"** technique. When the similarity measures are low and also SRM is low, the manual process is suggested for further approach. Because it may create some garbage knowledge. Besides, the weights in the new links are also calculated using equation 15 and it is stored in the updated ontology. At every merging the new weights are assigned to the edges. Here, if an edge in input ontology is already in output ontology then the edge in output shows more popularity of its uses among the service consumers. The more similar edges are imposed by the consumers; the more the weights become increased by the value α (user defined positive value for weight learning). By integrating consumer's concepts into ontology, service providers become more discoverable in semantic measure algorithms.

5. Experiments and Result Analysis

To check the performance of semantic discovery of web services, the proposed architecture is implemented. The implementation tools contain a private JUDDI server [14], Protégé ontology creator and validator and Jena APIs for ontology matching and alignment. A simulation test bed is created with a real-world OWL-S service retrieval test collection OWLS-TC v4 [15] to evaluate different aspects of the architecture.This collection contains services which are retrieved mainly from public IBM UDDI registries, and are semi-automatically transformed from WSDL to OWL-S. More specifically, it comprises a set of ontologies, derived from 9 different domains (education, medical care, food, travel, communication, economy, weapons, geography and simulation). Through the simulation the following questions are tried to be answer.

- Does the semantic search pin point to the desired web service whether syntactic search results noise data?

- Does the social learning improve discovery process whether tagging keywords result noise data?

To answer the questions, the discovery of "College" in 9 different domain of OWLS-TC each with 5 service provider's profile is simulated. The results are described in Fig 5(a). The Fig 5(a) depicts the semantic search as a winner as it has only 1 domain (education) response over threshold value 0.5 but syntactic search results 5 domain response (20 services) which is about 45% services of the simulation set up. Using Random weights in the ontologies the results can be more precise as it is seen in the Fig 5(a). In the later test, we take two Owls "College" as a service provider and "University" as the service requester. Then we try to improve "College" concept over time by contribution from different college like service consumers. And each time the matching with the original "University" is recorded. The final result is shown in the Fig 5(b). Here we see that after 10 social contributions the semantic matching has increased up to 39 times. But it again raises the question whether social learning creates noise response in semantic matching. To test that we run a simulation with the test data described in simulation set up using social learning on only one service which is called "College service". Then we take different service requester's requested Owls and match them with the "College" service after each social learning. For syntactic learning we just use "Tag Keywords Merging" and continue the N-Gram Matching techniques. The result of the simulation is described in Fig 5(c). Here we see that after 4 contributions, semantic search results 2% response over threshold value 0.5 but syntactic learning results 67% response rate. Thus the very lower response rate in semantic social learning ensures the discovery of desired web services whether higher response rate in syntactic techniques only produces noisy responses.

6. Conclusion

Service oriented architecture (SOA) is the face of new generation web and is going to experience an explosive growth because of it's automated nature and high productivity in information technology. The success of SOA hugely depends on efficient automatic discovery process which enables service invocation and composition automatically in consequences. SOA becomes more efficient if it can produce more accurate and desired discovery results. In this paper, a new architecture of SOA is proposed which enables semantic discovery of web services with an adaptive learning technique termed as social learning which produces more relevant discovery results over times. The ontology of the service provider improves it's concept through social concept contributions of service consumers and thus it becomes more discoverable over times. Through mathematical reasoning and experimental results it is shown that the concept matching and merging processes of the proposed architecture reduce noisy results of existing syntactic discovery processes significantly. The matching and merging processes in the architecture can be treated as a guideline for further researchers on adaptive learning in SOA. Besides, existing semantic discovery process may use the proposed procedures of matching and merging of ontology as a plug-in for integrating learning in it's core system to improve discovery results. Thus the proposed architecture should have significant importance in semantic discovery of web services. For future work necessary techniques should be incorporated with the architecture for automatic service invocation and composition of web services and thus the cycle of the dream sequences (automatic discovery, invocation and composition) of web services may get fulfilled.

References

[1] G. Alonso, F. Casati, H. Kuno, V. Machirajul, Web Services Concepts, Architectures and Applications, Springer, New York City, USA, 2003.
[2] N. Shadbolt, T. Berners-Lee, W. Hall, The Semantic Web Revisited, IEEE Intelligent Systems 1 (3) (2006) 96–101.
[3] K. Verma, A. P. Sheth, Semantically Annotating a Web Service, IEEE Internet Computing 11 (2) (2007) 83–85.
[4] A. A. Patil, S. A. Oundhakar, A. P. Sheth, K. Verma, Meteor-S Web Service Annotation Framework, in: Proceedings of the 13th international conference on World Wide Web'04, W3C, 2004, pp. 17–20.
[5] J. Cardoso, Discovering Semantic Web Services with and without a Common Ontology Commitment, in: Proceedings of the 3rd International Workshopon Semantic and Dynamic Web Processes, SDWP'06, IEEE Computer Society, 2006, pp. 183–190.
[6] J. Pathak, N. Koul, D. Caragea, V. Honavar, A Framework for Semantic Web Services Discovery, in: Proceedings of the 7th ACM International Workshop on Web Information and Data Management, IEEE Xplore, 2005, pp. 603–607.

[7] I. Mecar, A. Devlic, K. Trzec, Agent-oriented semantic discovery and mathcmaking of web services, in: Proceedings of the 8th International Conference on Telecommunications, ACM Press, 2005, pp. 45–50.
[8] B. Zhou, T. Huang, Semantic WEB Service Discovery Search with Ontology Learning, in: Proceedings of the 18th International Conference on Computer Science and Software Engineering, IEEE Xplore, 2008, pp. 151–155.
[9] K.Christensen, T.H.Olesen, L.L.Thomsen, Matching Semantically Described Web Services Using Ontologies, Technologija Journal on Information Technology And Control 35 (3) (2006) 267–275.
[10] N. Srinivasan, M. Paolucci, K. Sycara, Semantic Web Service Discovery in the OWL-S IDE, in: Proceedings of the 39th Annual Hawaii International Conference on System Sciences, ACM Press, 2006, pp. 109–111.
[11] J. Euzenat, J. Barrasa, P. Bouquet, R. Dieng, M. Ehrig, M. Hauswirth, M. Jarrar, R. Lara, D.Maynard, A. Napoli, G. Stamou, H. Stuckenschmidt, P. Shvaiko, S. Tessaris, S. van Acker, I. Zaihrayeu, T. L. Bach, State of the art on ontology alignment, Tech. rep., knowledge Web project [Online]. Available: http://knowledgeweb.semanticweb.org/. Last accessed 14th April 2011 (June 2004).
[12] F. Giunchiglia, P. Shvaiko, M. Yatskevich, S-Match: an algorithm and an implementation of semantic matching, in: Proceedings of the the European Semantic Web Symposium, IEEE Xplore, 2004, pp. 61–75.
[13] Bruijn, J. Deuke, State-of-the-art Survey on Ontology Merging and Aligning V1, Project report, SEKT (July 2003).
[14] JUDDI, Project Website of Java Universal Description Discovery integration, [Online]. Available: http://ws.apache.org/juddi/. Last accessed 14th April 2011.
[15] IBM UDDI, Web Service Technichal Chart OWLS-TC, [Online]. Available: http://www.semwebcentral.org/projects/owls-tc/. Last accessed 27th April 2010.